1. **Write a Java program to print the string data in initcap format?**

```
public class SampleTest {

    public static void main(String[] args) {
        String input = "hello world from java 8";
        String result = Arrays.stream(input.split(" "))
                              .map(word -> word.substring(0, 1).toUpperCase() + word.substring(1))
                              .collect(Collectors.joining(" "));
        System.out.println(result); // Output: Hello World From Java 8

    }
```

2. **How do you find frequency of each character in a string using Java 8 streams?**

```
public class SampleTest {

    public static void main(String[] args) {
        String inputString = "Java Concept Of The Day";

        Map<Character, Long> charCountMap =
                    inputString.chars()
                               .mapToObj(c -> (char) c)
                               .collect(Collectors.groupingBy(word -> word, Collectors.counting()));
        System.out.println(charCountMap); //{ =4, a=3, c=1, C=1, D=1, e=2, f=1, h=1, J=1, n=1, O=1, o=1, p=1, T=1, t=1, v=1, y=1}

    }
}
```

3. **How do you find frequency of each element in an array or a list?**

```
public class SampleTest {

    public static void main(String[] args) {

        List<String> stationeryList = Arrays.asList("Pen", "Eraser", "Note Book", "Pen", "Pencil", "Stapler", "Note Book", "Pencil");

        Map<String, Long> stationeryCountMap =
                stationeryList.stream().collect(Collectors.groupingBy(word -> word, Collectors.counting()));

        System.out.println(stationeryCountMap); //{Pen=2, Stapler=1, Pencil=2, Note Book=2, Eraser=1}
    }

}
```

4. **How do you sort the given list of decimals in reverse order?**

```
public class SampleTest {

    public static void main(String[] args) {
        List<Double> decimalList = Arrays.asList(12.45, 23.58, 17.13, 42.89, 33.78, 71.85, 56.98, 21.12);
        decimalList.stream().sorted(Comparator.reverseOrder()).forEach(System.out::print); //71.85 56.98 42.89 33.78 23.58 21.12 17.13 12.45
    }

}
```

5. **From the given list of integers, print the numbers which are multiples of 5?**

```
public class SampleTest {

    public static void main(String[] args) {
        List<Integer> listOfIntegers = Arrays.asList(45, 12, 56, 15, 24, 75, 31, 89);
        listOfIntegers.stream().filter(i -> i % 5 == 0).forEach(System.out::print); //45 15 75
    }

}
```

6. **Given a list of integers, find maximum and minimum of those numbers?**

```java
public class SampleTest {

    public static void main(String[] args) {
        List<Integer> listOfIntegers = Arrays.asList(45, 12, 56, 15, 24, 75, 31, 89);
        int max = listOfIntegers.stream().max(Comparator.naturalOrder()).get();
        System.out.println("Maximum Element : "+max); //Maximum Element : 89
        int min = listOfIntegers.stream().min(Comparator.naturalOrder()).get();
        System.out.println("Minimum Element : "+min); //Minimum Element : 12
    }

}
```

7. **How do you merge two unsorted arrays into single sorted array using Java 8 streams?**

```java
public class SampleTest {

    public static void main(String[] args) {
        int[] a = new int[] {4, 2, 7, 1};
        int[] b = new int[] {8, 3, 9, 5};
        int[] c = IntStream.concat(Arrays.stream(a), Arrays.stream(b)).sorted().toArray();
        System.out.println(Arrays.toString(c)); //[1, 2, 3, 4, 5, 7, 8, 9]
    }

}
```

8. **How do you merge two unsorted arrays into single sorted array without duplicates?**

```java
public class SampleTest {

    public static void main(String[] args) {
        int[] a = new int[] {4, 2, 5, 1};
        int[] b = new int[] {8, 1, 9, 5};
        int[] c = IntStream.concat(Arrays.stream(a), Arrays.stream(b)).sorted().distinct().toArray(); //[1, 2, 4, 5, 8, 9]
        System.out.println(Arrays.toString(c));
    }

}
```

9. **How do you get three maximum numbers and three minimum numbers from the given list of integers?**

```java
public class SampleTest {

    public static void main(String[] args) {
        List<Integer> listOfIntegers = Arrays.asList(45, 12, 56, 15, 24, 75, 31, 89);

        // 3 minimum Numbers
        listOfIntegers.stream().sorted().limit(3).forEach(System.out::print); //12 15 24

        // 3 Maximum Numbers
        listOfIntegers.stream().sorted(Comparator.reverseOrder()).limit(3).forEach(System.out::print); //89 75 56
    }

}
```

10. **Java 8 program to check if two strings are anagrams or not?**

```java
public class SampleTest {

    public static void main(String[] args) {
        String s1 = "RaceCar";
        String s2 = "CarRace";

        s1 = Stream.of(s1.split("")).map(String::toUpperCase).sorted().collect(Collectors.joining());

        s2 = Stream.of(s2.split("")).map(String::toUpperCase).sorted().collect(Collectors.joining());

        if (s1.equals(s2)) {
            System.out.println("Two strings are anagrams"); //Two strings are anagrams
        } else {
            System.out.println("Two strings are not anagrams");
        }
    }

}
```

11. Find sum of all digits of a number in Java 8?

```java
public class LogicalPrograms {

    public static void main(String[] args) {
        int i = 123456;
        System.out
                .println("Sum of all numbers : " + Arrays.stream(String.valueOf(i).split("")).collect(Collectors.summingInt(Integer::parseInt)));
        //Sum of all numbers : 21
    }

}
```

12. Find second largest number in an integer array?

```java
public class SampleTest {

    public static void main(String[] args) {
        List<Integer> listOfIntegers = Arrays.asList(45, 12, 56, 15, 24, 75, 31, 89);
        Integer secondLargestNumber = listOfIntegers.stream().sorted(Comparator.reverseOrder()).skip(1).findFirst().get();
        System.out.println(secondLargestNumber); //75
    }

}
```

13. Given a list of strings, sort them according to increasing order of their length?

```java
public class SampleTest {

    public static void main(String[] args) {
        List<String> listOfStrings = Arrays.asList("Java", "Python", "C#", "HTML", "Kotlin", "C++", "COBOL", "C");
        listOfStrings.stream().sorted(Comparator.comparing(String::length)).forEach(System.out::print);//C C# C++ Java HTML COBOL Python Kotlin
    }

}
```

14. Given an integer array, find sum and average of all elements?

```java
public class SampleTest {

    public static void main(String[] args) {
        int[] a = new int[] {45, 12, 56, 15, 24, 75, 31, 89};
        int sum = Arrays.stream(a).sum();
        System.out.println("Sum = "+sum); //Sum = 347
        double average = Arrays.stream(a).average().getAsDouble();
        System.out.println("Average = "+average); //Average = 43.375
    }

}
```

15. **How do you find common elements between two arrays?**

```java
public class SampleTest {

    public static void main(String[] args) {
        List<Integer> list1 = Arrays.asList(71, 21, 34, 89, 56, 28);
        List<Integer> list2 = Arrays.asList(12, 56, 17, 21, 94, 34);
        list1.stream().filter(list2::contains).forEach(System.out::print); //21 34 56
    }

}
```

16. **Reverse each word of a string using Java 8 streams?**

```java
public class SampleTest {

    public static void main(String[] args) {
          String str = "Java Concept Of The Day";
            String reversedStr = Arrays.stream(str.split(" "))
                                       .map(word -> new StringBuffer(word).reverse())
                                       .collect(Collectors.joining(" "));
            System.out.println(reversedStr); //avaJ tpecnoC fO ehT yaD
    }

}
```

17. **Reverse an integer array?**

```java
public class SampleTest {

    public static void main(String[] args) {
        Integer[] array = {1, 2, 3, 4, 5};
        List<Integer> list = Arrays.asList(array);
        Collections.reverse(list);
        Integer[] reversedArray = list.toArray(new Integer[0]);
        System.out.println(Arrays.toString(reversedArray)); //[5, 4, 3, 2, 1]

    }

}
```

18. **How do you find sum of first 10 natural numbers?**

```java
public class SampleTest {

    public static void main(String[] args) {
        int sum = IntStream.range(1, 11).sum();
        System.out.println(sum); //55

    }

}
```

19. Print first 10 even numbers?

```java
public class SampleTest {

    public static void main(String[] args) {
        IntStream.rangeClosed(1, 10).map(i -> i * 2).forEach(System.out::print); //2 4 6 8 10 12 14 16 18 20
    }
}
```

20. How do you find the most repeated element in an array?

```java
public class SampleTest {

    public static void main(String[] args) {
        List<String> listOfStrings = Arrays.asList("Pen", "Eraser", "Note Book", "Pen", "Pencil", "Pen", "Note Book", "Pencil");
        Map<String, Long> elementCountMap = listOfStrings.stream()
                                                .collect(Collectors.groupingBy(word -> word, Collectors.counting()));
        Entry<String, Long> mostFrequentElement = elementCountMap.entrySet().stream().max(Map.Entry.comparingByValue()).get();
        System.out.println("Most Frequent Element : "+mostFrequentElement.getKey()); //Most Frequent Element : Pen
        System.out.println("Count : "+mostFrequentElement.getValue()); //Count : 3
    }
}
```

21. Given a list of strings, find out those strings which start with a number?

```java
public class SampleTest {

    public static void main(String[] args) {
        List<String> listOfStrings = Arrays.asList("One", "2wo", "3hree", "Four", "5ive", "Six");
        listOfStrings.stream().filter(str -> Character.isDigit(str.charAt(0))).forEach(System.out::print);//2wo 3hree 5ive
    }
}
```

22. How do you extract duplicate elements from an array?

```java
public class SampleTest {

    public static void main(String[] args) {
        List<Integer> listOfIntegers = Arrays.asList(111, 222, 333, 111, 555, 333, 777, 222);
        Set<Integer> uniqueElements = new HashSet<>();
        Set<Integer> duplicateElements = listOfIntegers.stream().filter(i -> !uniqueElements.add(i))
                .collect(Collectors.toSet());
        System.out.println(duplicateElements);  //[333, 222, 111]
    }
}
```

23. Print duplicate characters in a string?

```java
public class SampleTest {

    public static void main(String[] args) {
        String inputString = "Java Concept Of The Day".replaceAll("\\s+", "").toLowerCase();
        Set<String> uniqueChars = new HashSet<>();
        Set<String> duplicateChars =
                Arrays.stream(inputString.split(""))
                        .filter(ch -> ! uniqueChars.add(ch))
                        .collect(Collectors.toSet());

        System.out.println(duplicateChars); //[a, c, t, e, o]
    }
}
```

24. Find first repeated character in a string?

```java
public class SampleTest {

    public static void main(String[] args) {
        String inputString = "Java Concept Of The Day".replaceAll("\\s+", "").toLowerCase();
        Map<String, Long> charCountMap =
                        Arrays.stream(inputString.split(""))
                                .collect(Collectors.groupingBy(word -> word, LinkedHashMap::new, Collectors.counting()));

        String firstRepeatedChar = charCountMap.entrySet()
                                               .stream()
                                               .filter(entry -> entry.getValue() > 1)
                                               .map(entry -> entry.getKey())
                                               .findFirst()
                                               .get();

        System.out.println(firstRepeatedChar); //a
    }
}
```

25. Find first non-repeated character in a string?

```java
public class SampleTest {

    public static void main(String[] args) {
        String inputString = "Java Concept Of The Day".replaceAll("\\s+", "").toLowerCase();
        Map<String, Long> charCountMap =
                        Arrays.stream(inputString.split(""))
                                .collect(Collectors.groupingBy(word -> word, LinkedHashMap::new, Collectors.counting()));

        String firstRepeatedChar = charCountMap.entrySet()
                                               .stream()
                                               .filter(entry -> entry.getValue() == 1)
                                               .map(entry -> entry.getKey())
                                               .findFirst()
                                               .get();

        System.out.println(firstRepeatedChar); //j
    }
}
```

26. How do you get last element of an array?

```java
public class SampleTest {

    public static void main(String[] args) {
        List<String> listOfStrings = Arrays.asList("One", "Two", "Three", "Four", "Five", "Six");
        String lastElement = listOfStrings.stream().skip(listOfStrings.size() - 1).findFirst().get();
        System.out.println(lastElement); //Six
    }
}
```

27. Find the age of a person in years if the birthday has given?

```java
public class SampleTest {

    public static void main(String[] args) {
        LocalDate birthDay = LocalDate.of(1985, 01, 23);
        LocalDate today = LocalDate.now();
        System.out.println(ChronoUnit.YEARS.between(birthDay, today)); //40
    }
}
```

28. **Given a list of integers, find out all the numbers starting with 1 using Stream functions?**

```java
public class SampleTest {

    public static void main(String[] args) {
        List<Integer> myList = Arrays.asList(10, 15, 8, 49, 25, 98, 32);
        myList.stream()
        .map(s -> s + "") // Convert integer to String
        .filter(s -> s.startsWith("1"))
        .forEach(System.out::println); //10 15

    }
}
```

29. **How to find duplicate elements in a given integers list in java using Stream functions?**

```java
public class SampleTest {

    public static void main(String[] args) {
        List<Integer> myList = Arrays.asList(10,15,8,49,25,98,98,32,15);
        myList.stream().distinct().forEach(noDuplicateData -> System.out.print(noDuplicateData)); //10 15 8 49 25 98 32
    }
}
```

30. **Given the list of integers, find the first element of the list using Stream functions?**

```java
public class SampleTest {

    public static void main(String[] args) {
        List<Integer> myList = Arrays.asList(10,15,8,49,25,98,98,32,15);
        myList.stream()
              .findFirst()
              .ifPresent(System.out::println); //10
    }
}
```

31. **Given a list of integers, find the total number of elements present in the list using Stream functions?**

```java
public class SampleTest {

    public static void main(String[] args) {
        List<Integer> myList = Arrays.asList(10,15,8,49,25,98,98,32,15);
        long count =  myList.stream()
                            .count();
        System.out.println(count); //9
    }
}
```

32. **Given a list of integers, find the maximum value element present in it using Stream functions?**

```
public class SampleTest {

    public static void main(String[] args) {
        int[] arr = {10,15,8,49,25,98,98,32,15};
         int maxdata = Arrays.stream(arr).boxed()
                 .max(Comparator.naturalOrder()).get();
         System.out.println(maxdata);    //98
    }
}
```

33. **Given an integer array nums, return true if any value appears at least twice in the array, and return false if every element is distinct?**

```
public class SampleTest {

    public static void main(String[] args) {
        int[] nums = {1,2,3,1};
        Set<Integer> setData = new HashSet<>();
        boolean flag =  Arrays.stream(nums)
                        .anyMatch(num -> !setData.add(num));
        System.out.println(flag); //true
    }
}
```

34. **Write a Java 8 program to concatenate two Streams?**

```
public class SampleTest {

    public static void main(String[] args) {
        List<String> list1 = Arrays.asList("Java", "8");
        List<String> list2 = Arrays.asList("explained", "through", "programs");
        Stream<String> concatStream = Stream.concat(list1.stream(), list2.stream());
        concatStream.forEach(str -> System.out.print(str + " ")); //Java 8 explained through programs
    }
}
```

35. **How to find only duplicate elements with its count from the String ArrayList in Java8?**

```
public class SampleTest {

    public static void main(String[] args) {
        List<String> names = Arrays.asList("AA", "BB", "AA", "CC");
        Map<String, Long> namesCount = names.stream()
                .collect(Collectors.groupingBy(word -> word, Collectors.counting()))
                .entrySet()
                .stream()
                .filter(entry -> entry.getValue() > 1)
                .collect(Collectors.toMap(Map.Entry::getKey, Map.Entry::getValue));
        System.out.println(namesCount); //{AA=2}
    }
}
```

36. In Given string find the max and min length word?

```
public class SampleTest {

    public static void main(String[] args) {
        String data = "I am learning streams api in java";

        String maxLengthWord =  Arrays.stream(data.split(" ")).max(Comparator.comparing(String::length)).get();
        System.out.println("maxLengthWord : " + maxLengthWord); //maxLengthWord : learning
        String minLengthWord =  Arrays.stream(data.split(" ")).min(Comparator.comparing(String::length)).get();
        System.out.println("minLengthWord : " + minLengthWord); //minLengthWord : I
    }
}
```

37. Remove duplicates from the string and return in the same order

```
public class SampleTest {

    public static void main(String[] args) {
        String data = "dabcadefg";
        Arrays.stream(data.split("")).distinct().forEach(System.out::print); //dabcefg
    }
}
```

38. Find the word that has the second and third highest length words?

```
public class SampleTest {

    public static void main(String[] args) {
        String data = "I am learning streams api in java";

        //Second highest word

        String secondHighestLengthWord = Arrays.stream(data.split(" ")).sorted(Comparator.comparing(String::length).reversed()).skip(1).findFirst().get();
        System.out.println("secondHighestLengthWord : " + secondHighestLengthWord); //secondHighestLengthWord : streams

        //Third Higest word
        String thirdHighestLengthWord = Arrays.stream(data.split(" ")).sorted(Comparator.comparing(String::length).reversed()).skip(2).findFirst().get();
        System.out.println("thirdHighestLengthWord : " + thirdHighestLengthWord); // thirdHighestLengthWord : java
    }
}
```

39. Divide given integer list into lists of even and odd numbers?

```
public class SampleTest {

    public static void main(String[] args) {
        int [] arr = {1,2,3,4,5,67,8};

        List<Integer> numsList =  Arrays.stream(arr).boxed().toList();
        List<List<Integer>> convertedList = numsList.stream().collect(Collectors.groupingBy(num -> num%2==0, Collectors.toList()))
                .entrySet().stream().map(data -> data.getValue()).toList();
        System.out.println(convertedList); //[[1, 3, 5, 67], [2, 4, 8]]
    }
}
```

40. Given an array, find the sum of unique elements?

```
public class SampleTest {
    public static void main(String[] args) {
        int[] arr = { 1, 2, 3, 4, 5, 67, 8 };
        System.out.println(Arrays.stream(arr).sum()); // 90
    }
}
```

41. Given a string, find the first non-repeated character?

```
public class SampleTest {
    public static void main(String[] args) {
        String text = "Hello";

        String firstNonRepeatedChar = Arrays.stream(text.split("")).collect(Collectors.groupingBy(word -> word,LinkedHashMap::new,Collectors.counting()))
                .entrySet().stream().filter(map -> map.getValue()==1).map(data -> data.getKey()).findFirst().get();
        System.out.println("firstNonRepeatedChar : " + firstNonRepeatedChar); //firstNonRepeatedChar : H

    }
}
```

42. Given a string, find the first repeated character?

```
public class Important {

    public static void main(String[] args) {
        String data = "Hello World";

        Character firstRepeatedCharacter = data.chars().mapToObj(c -> (char)c).collect(Collectors.groupingBy(word -> word, LinkedHashMap::new,Collectors.counting())).
                            entrySet().stream().filter(map -> map.getValue()>1).findFirst().get().getKey();
        System.out.println("firstRepeatedCharacter : " + firstRepeatedCharacter); //firstRepeatedCharacter : l
    }

}
```

43. Given a list of strings, create a list that contains only integers?

```
public class Important {

    public static void main(String[] args) {
        String[] data = {"abc","123","xyz","456"};

        List<Integer> list = Arrays.stream(data).filter(item -> item.matches("[0-9]+")).map(Integer::valueOf).toList();
        System.out.println("Numbers seperated list : "+ list); //Numbers seperated list : [123, 456]
    }

}
```

44. Find the products of the first two elements in an array?

```
public class Important {

    public static void main(String[] args) {
        int[] arr = {12,7,8,9};

        Integer productOfTwoNumbers = Arrays.stream(arr).boxed().toList().stream().limit(2).reduce(1,(a,b)-> a*b);
        System.out.println("productOfTwoNumbers : " + productOfTwoNumbers); //productOfTwoNumbers : 84
    }

}
```

45. Group /Pair anagrams from a list of Strings?

```
public class Important {

    public static void main(String[] args) {
        String[] arr = {"pat","tap","pan","nap","tree"};

        Collection<List<String>> anagrams = Arrays.stream(arr).toList().stream().collect(Collectors.groupingBy(data -> Arrays.stream(data.toLowerCase().split("")).sorted().toList())).values();
        System.out.println("anagrams : " + anagrams); //anagrams : [[pan, nap], [pat, tap], [tree]]
    }

}
```

46. Given the string[] group the strings based on the middle character?

```
public class Important {

    public static void main(String[] args) {
        String[] arr = {"ewe","jji","jhj","kwk","aha"};
        System.out.println("Grouping the strings based on middle character : " + Arrays.stream(arr).toList().stream().collect(Collectors.groupingBy(data -> data.substring(1, 2))));
        //Grouping the strings based on middle character : {w=[ewe, kwk], h=[jhj, aha], j=[jji]}
    }
}
```

47. Write a program to do the union of two integers arrays?

```
public class important {

    public static void main(String[] args) {
        List<Integer> list1 = Arrays.asList(1,2,3,4,5);
        List<Integer> list2 = Arrays.asList(6,7,8,9,10);

        System.out.println("Union of two integer lists is : " + Stream.concat(list1.stream(), list2.stream()).toList()); //Union of two integer lists is : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
    }
}
```

48. Remove all non-numeric characters from a list?

```
public class Important {

    public static void main(String[] args) {
        List<String> list1 = Arrays.asList("a1b12c","1a2b3");

        Pattern pattern = Pattern.compile("[^0-9]");

        System.out.println("After removing all non numeric characters : "+list1.stream().
                map(data -> pattern.matcher(data).replaceAll("")).toList()); //After removing all non numeric characters : [112, 123]
    }
}
```

49. Write a program to print only characters?

```
public class Important {

    public static void main(String[] args) {
        List<String> list1 = Arrays.asList("a1b12c","1a2b3","axg$%5");

        Pattern pattern = Pattern.compile("[^a-zA-Z]");

        System.out.println("After removing all non numeric characters : "+list1.stream().
                map(data -> pattern.matcher(data).replaceAll("")).toList()); //After removing all non numeric characters : [abc, ab, axg]
    }
}
```

50. Find the intersection of two lists using Java streams

```
public class Important {

    public static void main(String[] args) {
        List<Integer> list1 = Arrays.asList(1,3,5,4);
        List<Integer> list2 = Arrays.asList(1,3,6,9);

        System.out.println("Common Elements between two arrays is : " +
        list1.stream().filter(list2::contains).toList()); //Common Elements between two arrays is : [1, 3]
    }
}
```

51. Write a program to Multiply array elements

```
public class Important {

    public static void main(String[] args) {
        List<Integer> list1 = Arrays.asList(1,3,5,4);
        Integer  multiplication = list1.stream().reduce((a,b) -> a*b).get();
        System.out.println("Multiplication of elements in Array is : "+multiplication); //Multiplication of elements in Array is : 60
    }
}
```

52. Convert a list of string to uppercase and then concatenate

```java
public class Important {

    public static void main(String[] args) {
        List<String> list1 = Arrays.asList("a","b","c","d");
        String  concatinatedStringAfterUpperCase = list1.stream().map(String::toUpperCase).reduce((a,b) -> a+b).get();
        System.out.println("concatinatedStringAfterUpperCase is : "+concatinatedStringAfterUpperCase); //concatinatedStringAfterUpperCase is : ABCD
    }
}
```

53. Find the average salary from each department

```java
public class Employee {
     String name;
     Integer salary;
     String dept;

     public Employee(String name, Integer salary, String dept) {
         super();
         this.name = name;
         this.salary = salary;
         this.dept = dept;
     }
     public String getName() {
         return name;
     }
     public void setName(String name) {
         this.name = name;
     }
     public Integer getSalary() {
         return salary;
     }
     public void setSalary(Integer salary) {
         this.salary = salary;
     }
     public String getDept() {
         return dept;
     }
     public void setDept(String dept) {
         this.dept = dept;
     }

}
```

```java
public class Important {

    public static void main(String[] args) {
        List<Employee> empList = new ArrayList<>();
        empList.add(new Employee("Alice", 5000,"IT"));
        empList.add(new Employee("Bob", 3500,"Testing"));
        empList.add(new Employee("Charlie", 7000,"IT"));
        empList.add(new Employee("Daid", 4500,"Testing"));

        Map<String, Double> avergeSalaryByDepartment = empList.stream().collect(
                Collectors.groupingBy(Employee::getDept,Collectors.averagingInt(Employee::getSalary)));

        avergeSalaryByDepartment.forEach((dept,sal)->System.out.println(dept + " : " + sal)); //IT : 6000.0 Testing : 4000.0
    }
}
```

54. Write a program on Group By Functions?

```java
public class Employee {
    private int employeeId;
    private String name;
    private String gender;
    private double salary;
    public Employee(int employeeId, String name, String gender, double salary) {
        super();
        this.employeeId = employeeId;
        this.name = name;
        this.gender = gender;
        this.salary = salary;
    }
    public int getEmployeeId() {
        return employeeId;
    }
    public void setEmployeeId(int employeeId) {
        this.employeeId = employeeId;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getGender() {
        return gender;
    }
    public void setGender(String gender) {
        this.gender = gender;
    }
    public double getSalary() {
        return salary;
    }
    public void setSalary(double salary) {
        this.salary = salary;
    }
    @Override
    public String toString() {
        return "Employee [employeeId=" + employeeId + ", name=" + name + ", gender=" + gender + ", salary=" + salary
                + "]";
    }
```

```java
public class EmployeeGroupBY {

    public static void main(String[] args) {
        List<Employee> empList = new ArrayList<>();
        empList.add(new Employee(101,"Alex","Male",20000));
        empList.add(new Employee(102,"Bob","Male",30000));
        empList.add(new Employee(105,"Sita","FeMale",80000));
        empList.add(new Employee(104,"Carey","Male",60000));
        empList.add(new Employee(103,"Geeta","FeMale",10000));

        //Group By Employee Name
        Map<String, List<Employee>> groupByEmployeeName = empList.stream().collect(Collectors.groupingBy(Employee::getName));
        System.out.println("groupByEmployeeName : "+ groupByEmployeeName);
        //groupByEmployeeName : {Alex=[Employee [employeeId=101, name=Alex, gender=Male, salary=20000.0]], Geeta=[Employee [employeeId=103, name=Geeta, gender=FeMale, salary=10000.0]],
        //Sita=[Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0]], Bob=[Employee [employeeId=102, name=Bob, gender=Male, salary=30000.0]],
        //Carey=[Employee [employeeId=104, name=Carey, gender=Male, salary=60000.0]]}

        //Logic to print employees based on Gender
        Map<String, List<String>> groupByGenderAndName = empList.stream()
                .collect(Collectors.groupingBy(Employee::getGender,Collectors.mapping(Employee::getName, Collectors.toList())));
        System.out.println("groupByGenderAndName : " + groupByGenderAndName); //groupByGenderAndName : {Male=[Alex, Bob, Carey], FeMale=[Sita, Geeta]}

        //Logic to count the employees based on gender
        Map<String, Long> employeesCountBasedOnGender = empList.stream().collect(Collectors.groupingBy(Employee::getGender,Collectors.counting()));
        System.out.println("employeesCountBasedOnGender : " + employeesCountBasedOnGender); //employeesCountBasedOnGender : {Male=3, FeMale=2}

        //Write logic to find the average salary based on gender
        Map<String, Double> genderWiseAverageSalary = empList.stream().collect(Collectors.groupingBy(Employee::getGender,Collectors.averagingDouble(Employee::getSalary)));
        System.out.println("genderWiseAverageSalary : "+ genderWiseAverageSalary); //genderWiseAverageSalary : {Male=36666.666666666664, FeMale=45000.0}



    }

}
```

55. Write few programs on Java 8 Streams on Employee Object?

```java
public class Employee {
    private int employeeId;
    private String name;
    private String gender;
    private double salary;
    public Employee(int employeeId, String name, String gender, double salary) {
        super();
        this.employeeId = employeeId;
        this.name = name;
        this.gender = gender;
        this.salary = salary;
    }
    public int getEmployeeId() {
        return employeeId;
    }
    public void setEmployeeId(int employeeId) {
        this.employeeId = employeeId;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getGender() {
        return gender;
    }
    public void setGender(String gender) {
        this.gender = gender;
    }
    public double getSalary() {
        return salary;
    }
    public void setSalary(double salary) {
        this.salary = salary;
    }
    @Override
    public String toString() {
        return "Employee [employeeId=" + employeeId + ", name=" + name + ", gender=" + gender + ", salary=" + salary
                + "]";
    }
}
public class EmployeeStreamOperations {
    public static void main(String[] args) {
        List<Employee> empList = new ArrayList<>();
        empList.add(new Employee(101,"Alex","Male",20000));
        empList.add(new Employee(102,"Bob","Male",30000));
        empList.add(new Employee(105,"Sita","FeMale",80000));
        empList.add(new Employee(104,"Carey","Male",60000));
        empList.add(new Employee(103,"Geeta","FeMale",10000));
        empList.add(new Employee(105,"Sita","FeMale",80000));

        //Write logic to display distinct Employee details in Ascending Order
        TreeSet<Employee> uniqueEmployeeList = empList.stream().collect(Collectors.toCollection(() -> new TreeSet<>(Comparator.comparing(Employee::getEmployeeId))));
        System.out.println("uniqueEmployeeList : "+ uniqueEmployeeList);
        //uniqueEmployeeList : [Employee [employeeId=101, name=Alex, gender=Male, salary=20000.0], Employee [employeeId=102, name=Bob, gender=Male, salary=30000.0],
        //Employee [employeeId=103, name=Geeta, gender=FeMale, salary=10000.0], Employee [employeeId=104, name=Carey, gender=Male, salary=60000.0],
        //Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0]]

        //Write logic to display distinct Employee details in Descending Order
        TreeSet<Employee> uniqueEmployeeListInDescendingOrder = empList.stream().collect(Collectors.toCollection(
                () -> new TreeSet<>(Comparator.comparing(Employee::getEmployeeId).reversed())));
        System.out.println("uniqueEmployeeListInDescendingOrder : " + uniqueEmployeeListInDescendingOrder);
        //uniqueEmployeeListInDescendingOrder : [Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0],
        //Employee [employeeId=104, name=Carey, gender=Male, salary=60000.0], Employee [employeeId=103, name=Geeta, gender=FeMale, salary=10000.0],
        //[employeeId=102, name=Bob, gender=Male, salary=30000.0], Employee [employeeId=101, name=Alex, gender=Male, salary=20000.0]]

        //Write a logic to sort employee based on his salary in ascending order
        List<Employee> sortingEmployeeBySalary = empList.stream().sorted(Comparator.comparing(Employee::getSalary)).toList();
        System.out.println("sortingEmployeeBySalary : "+ sortingEmployeeBySalary);
        //sortingEmployeeBySalary : [Employee [employeeId=103, name=Geeta, gender=FeMale, salary=10000.0],
        //Employee [employeeId=101, name=Alex, gender=Male, salary=20000.0], Employee [employeeId=102, name=Bob, gender=Male, salary=30000.0],
        //Employee [employeeId=104, name=Carey, gender=Male, salary=60000.0], Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0],
        //Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0]]

        //Write a logic to sort employee based on his salary in descending order
        List<Employee> sortEmployeeBySalaryInDescending = empList.stream().sorted(Comparator.comparing(Employee::getSalary).reversed()).toList();
        System.out.println("sortEmployeeBySalaryInDescending : " + sortEmployeeBySalaryInDescending);
        //sortEmployeeBySalaryInDescending : [Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0],
        //Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0], Employee [employeeId=104, name=Carey, gender=Male, salary=60000.0],
        //Employee [employeeId=102, name=Bob, gender=Male, salary=30000.0], Employee [employeeId=101, name=Alex, gender=Male, salary=20000.0],
        //Employee [employeeId=103, name=Geeta, gender=FeMale, salary=10000.0]]

    }
```

```java
//Filter Employees whose salary is >20000 and display their names
List<String> employeeSalaryGreaterThan20000 = empList.stream().filter(emp -> emp.getSalary()>20000).map(emp->emp.getName()).toList();
System.out.println("employeeSalaryGreaterThan20000 : "+ employeeSalaryGreaterThan20000); //employeeSalaryGreaterThan20000 : [Bob, Sita, Carey, Sita]

//Filter employee list whose name starts with S
List<Employee> employeeNameStartsWithS = empList.stream().filter(emp -> emp.getName().startsWith("S")).toList();
System.out.println("employeeNameStartsWithS : "+ employeeNameStartsWithS);
//employeeNameStartsWithS : [Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0], Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0]]

//Find the Max salary Employee details
Employee employeeWithMaxSalary = empList.stream().max(Comparator.comparing(Employee::getSalary)).get();
System.out.println("employeeWithMaxSalary : "+ employeeWithMaxSalary); //employeeWithMaxSalary : Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0]

//Find Employee with Minimum Salary
Employee minimumSalaryEmployee = empList.stream().min(Comparator.comparing(Employee::getSalary)).get();
System.out.println("minimumSalaryEmployee : "+ minimumSalaryEmployee); //minimumSalaryEmployee : Employee [employeeId=103, name=Geeta, gender=FeMale, salary=10000.0]

//Second Highest Salary
Employee secondHighestSalaryEmployee = empList.stream().distinct().sorted(Comparator.comparing(Employee::getSalary).reversed()).skip(1).findFirst().get();
System.out.println("secondHighestSalaryEmployee : "+ secondHighestSalaryEmployee); //secondHighestSalaryEmployee : Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0]

//Display the employee names whose salary between 30000 and 50000
List<String> employeeNamesList = empList.stream().filter(emp -> emp.getSalary()>=30000 && emp.getSalary()<=80000).map(emp -> emp.getName()).toList();
System.out.println("employeeNamesList : "+ employeeNamesList); //employeeNamesList : [Bob, Sita, Carey, Sita]

//Count the employees whose salary between 30000 and 50000
long employeeCount = empList.stream().filter(emp -> emp.getSalary()>=30000 && emp.getSalary()<=80000).count();
System.out.println("employeeCount : "+ employeeCount); //employeeCount : 4

//Write a logic to check any of the employee name is Alex
boolean anyMatch = empList.stream().anyMatch(emp -> emp.getName().equalsIgnoreCase("Alex"));
System.out.println("anyMatch : "+ anyMatch); //anyMatch : true
```

56. Write Java 8 programs to perform possible operation on Strings?

```java
public class Java8StringsOperations {
    public static void main(String[] args) {
        //Display text in init cap format
        String text = "welcome to hello world hello";
        String initCapText = Arrays.stream(text.split(" ")).map(data -> data.substring(0,1).toUpperCase()+data.substring(1)).collect(Collectors.joining(" "));
        System.out.println("initCapText : " + initCapText); //initCapText : Welcome To Hello World

        //Reverse the words by preserving the spaces
        String reverseWordsByPreservingSpaces = Arrays.stream(text.split(" ")).map(data -> new StringBuilder(data).reverse()).collect(Collectors.joining(" "));
        System.out.println("reverseWordsByPreservingSpaces : "+ reverseWordsByPreservingSpaces); //reverseWordsByPreservingSpaces : emoclew ot olleh dlrow

        //Write logic to find count of each character with out spaces
        String newText = text.replace(" ", "");
        Map<Character, Long> characterItsCountWithOutSpaces = newText.chars().mapToObj(ch -> (char)ch)
                .collect(Collectors.groupingBy(Function.identity(),Collectors.counting()));
        System.out.println("characterItsCountWithOutSpaces : "+ characterItsCountWithOutSpaces);
        //characterItsCountWithOutSpaces : {r=1, c=1, d=1, t=1, e=3, w=2, h=1, l=4, m=1, o=4}

        //Write a logic to find the words count
        Map<String, Long> wordsCountMap = Arrays.stream(text.split(" ")).collect(Collectors.groupingBy(Function.identity(),Collectors.counting()));
        System.out.println("wordsCountMap : " + wordsCountMap); //wordsCountMap : {world=1, hello=1, to=1, welcome=1}

        //Logic to find Second largest word
        String data = "Hello Worlds Hi";
        String secondHighestLengthWord = Arrays.stream(data.split(" ")).sorted(Comparator.comparing(String::length).reversed()).skip(1).findFirst().get();
        System.out.println("secondHighestLengthWord : " + secondHighestLengthWord); //secondHighestLengthWord : Hello

        //Max Length word
        String maxLengthWord = Arrays.stream(text.split(" ")).max(Comparator.comparing(String::length)).get();
        System.out.println("maxLengthWord : "+ maxLengthWord); //maxLengthWord : welcome

        //Min Length word
        String minLengthWord = Arrays.stream(text.split(" ")).min(Comparator.comparing(String::length)).get();
        System.out.println("minLengthWord : "+ minLengthWord); //minLengthWord : to
```

```java
    //Check two strings are anagrams
    String word1 = "abc";
    String word2 = "bac";
    String sortedWord1 = Arrays.stream(word1.split("")).map(String::toUpperCase).sorted().collect(Collectors.joining());
    String sortedWord2 = Arrays.stream(word2.split("")).map(String::toUpperCase).sorted().collect(Collectors.joining());
    boolean isAnagram = sortedWord1.equals(sortedWord2)?true:false;
    System.out.println("Both are anagrams : "+ isAnagram); //Both are anagrams : true

    //list of words sort them by length
    String sortedBasedOnLength = Arrays.stream(text.split(" ")).sorted(Comparator.comparing(String::length)).collect(Collectors.joining(" "));
    System.out.println("sortedBasedOnLength : "+ sortedBasedOnLength);  //sortedBasedOnLength : to hello world welcome

    //First repeated character
    String spaceReplacedText = text.replace(" ", "");
     Character firstRepeatedChar = spaceReplacedText.chars().mapToObj(ch -> (char)ch).collect(Collectors.groupingBy(Function.identity(),LinkedHashMap::new,Collectors.counting()))
    .entrySet().stream().filter(value -> value.getValue()>1).map(item -> item.getKey()).findFirst().get();
    System.out.println("firstRepeatedChar :" + firstRepeatedChar); //firstRepeatedChar :w

    //First non repeated character
    Character firstNonRepearedChar = spaceReplacedText.chars().mapToObj(ch -> (char)ch).collect(Collectors.groupingBy(Function.identity(),LinkedHashMap::new,Collectors.counting()))
            .entrySet().stream().filter(val -> val.getValue()==1).map(key -> key.getKey()).findFirst().get();
    System.out.println("firstNonRepearedChar : "+ firstNonRepearedChar); //firstNonRepearedChar : c

    //Max Repeated Word
    LinkedHashMap<Character, Long> countingMap = spaceReplacedText.chars().mapToObj(ch -> (char)ch).collect(Collectors.groupingBy(Function.identity(),LinkedHashMap::new,Collectors.counting()));
    Character maxRepeatedCharacter = countingMap.entrySet().stream().max(Map.Entry.comparingByValue()).get().getKey();
    System.out.println("maxRepeatedCharacter : "+ maxRepeatedCharacter); //maxRepeatedCharacter : l

    //name starts with h
    String namesSartsWithSIs = Arrays.stream(text.split(" ")).filter(word -> word.startsWith("h")).collect(Collectors.joining(" "));
    System.out.println("namesSartsWithSIs : "+ namesSartsWithSIs); //namesSartsWithSIs : hello hello


//Pattern Examples
String specialChardData = "abc 1abc #abc @abc 2bcd bcd";

//Print only characters
Pattern charsPattern = Pattern.compile("[^a-zA-Z]");
String onlyCharacters = Arrays.stream(specialChardData.split(" ")).map(charsData -> charsPattern.matcher(charsData).replaceAll("")).collect(Collectors.joining(" "));
System.out.println("onlyCharacters : "+ onlyCharacters); //onlyCharacters : abc abc abc abc bcd bcd

//Print only numbers
Pattern onlyNumbersPattern = Pattern.compile("[^0-9]");
String onlyNumbersData = Arrays.stream(specialChardData.split(" ")).map(onlyNums -> onlyNumbersPattern.matcher(onlyNums).replaceAll("")).collect(Collectors.joining(" "));
System.out.println("onlyNumbersData : "+ onlyNumbersData); //onlyNumbersData :  1   2

//Print Alpha numerics
Pattern alphaNumericPattern = Pattern.compile("[^a-zA-Z0-9]");
String alphaNumericData = Arrays.stream(specialChardData.split(" ")).map(alphaNumeric -> alphaNumericPattern.matcher(alphaNumeric).replaceAll("")).collect(Collectors.joining(" "));
System.out.println("alphaNumericData : "+ alphaNumericData); //alphaNumericData : abc 1abc abc abc 2bcd bcd

//Print only special characters
Pattern specialCharactersPattern = Pattern.compile("[a-zA-Z0-9]+");
String specialCharactersOnly = Arrays.stream(specialChardData.split(" ")).map(specialChars -> specialCharactersPattern.matcher(specialChars).replaceAll("")).collect(Collectors.joining(" "));
System.out.println("specialCharactersOnly : "+ specialCharactersOnly); //specialCharactersOnly :   # @

//print only duplicate words in String
Set<String> duplicateSet = new LinkedHashSet<>();
String duplicateCharactersOnly = Arrays.stream(text.split(" ")).filter(charData -> !duplicateSet.add(charData)).collect(Collectors.joining(" "));
System.out.println("Duplicate Characters : "+ duplicateCharactersOnly); //Duplicate Characters : hello

//remove duplicated from string and return in same order
String uniqueWOrdsInAscendingOrder = Arrays.stream(text.split(" ")).distinct().sorted().collect(Collectors.joining(" "));
System.out.println("uniqueWOrdsInAscendingOrder : "+ uniqueWOrdsInAscendingOrder); //uniqueWOrdsInAscendingOrder : hello to welcome world
```

57. Java 8 programs on integers?

```java
public class StreamIntegersPrograms {

    public static void main(String[] args) {
        int data =123456;

        //Calculate the sum of all numbers
        Integer sumOfAllIntegers = String.valueOf(data).chars().
                mapToObj(ch -> String.valueOf((char)ch)).collect(Collectors.summingInt(Integer::parseInt));
        System.out.println("Sum of all nymbers : " + sumOfAllIntegers); //Sum of all nymbers : 21

        //Reverse Number

        StringBuilder reverseNumber = new StringBuilder(String.valueOf(data).chars().
                mapToObj(ch -> String.valueOf((char)ch)).collect(Collectors.joining())).reverse();
        System.out.println("reverseNumber : "+ reverseNumber); //reverseNumber : 654321

    }

}
```

58. Few Java 8 programs on Arrays?

```java
public class StreamsArrayOperations {

    public static void main(String[] args) {

        int[] arr = {10,20,19,-1,0,59,-1,0};

        //Max Number
        Integer maxNumber = Arrays.stream(arr).boxed().max(Comparator.naturalOrder()).get();
        System.out.println("Max Number is Array Is : " + maxNumber); //Max Number is Array Is : 59

        //Min Number
        Integer minimumNumber = Arrays.stream(arr).boxed().min(Comparator.naturalOrder()).get();
        System.out.println("Min Number is Array Is : " + minimumNumber); //Min Number is Array Is : -1

        //Second highest number
        Integer secondHighestNumber = Arrays.stream(arr).boxed().sorted(Comparator.reverseOrder()).skip(1).findFirst().get();
        System.out.println("secondHighestNumber : "+ secondHighestNumber); //secondHighestNumber : 20

        //Find Last Number in Array
        Integer lastNumberInArray = Arrays.stream(arr).boxed().skip(arr.length-1).findFirst().get();
        System.out.println("lastNumberInArray : "+ lastNumberInArray); //lastNumberInArray : 59

        //Distinct Numbers
        List<Integer> uniqueNumberList = Arrays.stream(arr).boxed().distinct().toList();
        System.out.println("uniqueNumberList : "+ uniqueNumberList); //uniqueNumberList : [10, 20, 19, -1, 0, 59]

        //Only Duplicate Numbers
        Set<Integer> duplicateSet = new LinkedHashSet<>();
        System.out.println("Only Duplicate Elements : "+ Arrays.stream(arr).boxed().filter(num -> !duplicateSet.add(num)).toList()); //Only Duplicate Elements : [-1, 0]

        //In between Numbers
        List<Integer> inBetweenNumbers = Arrays.stream(arr).boxed().sorted().filter(data -> data>=10 && data<=50).toList();
        System.out.println("inBetweenNumbers : "+ inBetweenNumbers); //inBetweenNumbers : [10, 19, 20]

        //Sum In between numbers
        Integer sumOfInBetweenNumbers = Arrays.stream(arr).boxed().sorted().filter(data -> data>=10 && data<=50).collect(Collectors.summingInt(Integer::intValue));
        System.out.println("sumOfInBetweenNumbers : "+ sumOfInBetweenNumbers); //sumOfInBetweenNumbers : 49

        //Average of in between numbers
        Double averageOfInbetweenNumbers = Arrays.stream(arr).boxed().sorted().filter(data -> data>=10 && data<=50).collect(Collectors.averagingInt(Integer::intValue));
        System.out.println("averageOfInbetweenNumbers : "+ averageOfInbetweenNumbers); //averageOfInbetweenNumbers : 16.333333333333332

        //Display words in ascending order
        String [] stringArr = {"abcde","bace","cabt","aacm","abc"};
        List<String> ascendingOrderList = Arrays.stream(stringArr).sorted().toList();
        System.out.println("ascendingOrderList : "+ ascendingOrderList); //ascendingOrderList : [aac, abc, abc, bac, cab]

        //Display words in descending order
        List<String> descendingOrderList = Arrays.stream(stringArr).sorted(Comparator.reverseOrder()).toList();
        System.out.println("descendingOrderList : "+ descendingOrderList);//descendingOrderList : [cab, bac, abc, abc, aac]

        //Second highest word based on length
        String secondHighestLengthWord = Arrays.stream(stringArr).sorted(Comparator.comparing(String::length)).skip(1).findFirst().get();
        System.out.println("secondHighestLengthWord : "+ secondHighestLengthWord); //secondHighestLengthWord : bace

        //Concate two string arrays
        String [] anotherArray = {"123","456","aacm","abc"};
        List<String> concatenatedList = Stream.concat(Arrays.stream(stringArr), Arrays.stream(anotherArray)).toList();
        System.out.println("concatenatedList : "+ concatenatedList); //concatenatedList : [abcde, bace, cabt, aacm, abc, 123, 456, aacm, abc]

        //Common words in two arrays
        List<String> commonElementsList = Arrays.asList(stringArr).stream().filter(Arrays.asList(anotherArray)::contains).toList();
        System.out.println("commonElementsList : "+ commonElementsList); //commonElementsList : [aacm, abc]

        //Display only letters in list
        String[] allCharsArr = {"a12","12b","c23$","#43d"};
        Pattern onlyCharsPattern = Pattern.compile("[^a-zA-Z]");
        List<String> onlyCharsList = Arrays.stream(allCharsArr).map(data-> onlyCharsPattern.matcher(data).replaceAll("")).toList();
        System.out.println("onlyCharsList : "+ onlyCharsList); //onlyCharsList : [a, b, c, d]

        //Display only numbers in list
        Pattern onlyNumbersPattern = Pattern.compile("[^0-9]");
        List<String> onlyNumbersList = Arrays.stream(allCharsArr).map(data -> onlyNumbersPattern.matcher(data).replaceAll("")).toList();
        System.out.println("onlyNumbersList : "+ onlyNumbersList); //onlyNumbersList : [12, 12, 23, 43]

        //Display only alpha numerics in list
        Pattern alphaNumericPattern = Pattern.compile("[^a-zA-Z0-9]");
        List<String> alphaNumericList = Arrays.stream(allCharsArr).map(data -> alphaNumericPattern.matcher(data).replaceAll("")).toList();
        System.out.println("alphaNumericList : "+ alphaNumericList); //alphaNumericList : [a12, 12b, c23, 43d]

        //Display only special characters in list
        Pattern allCharsPattern = Pattern.compile("[a-zA-Z0-9]+");
        List<String> specialCharsList = Arrays.stream(allCharsArr).map(data -> allCharsPattern.matcher(data).replaceAll("")).toList();
        System.out.println("specialCharsList : " + specialCharsList); //specialCharsList : [, , $, #]


        int[] array1 = {1, 2, 3};
        int[] array2 = {4, 5, 6};

        //Summing up all the elements in array
        int sumOfAllElementsInArray = IntStream.concat(Arrays.stream(array1), Arrays.stream(array2)).sum();
        System.out.println("sumOfAllElementsInArray : "+ sumOfAllElementsInArray); //sumOfAllElementsInArray : 21

        // Summing up the same position elements in Array
        int[] summedArray = IntStream.range(0, array1.length)
                                     .map(i -> array1[i] + array2[i])
                                     .toArray();
        System.out.println("Summed Array: " + Arrays.toString(summedArray)); //Summed Array: [5, 7, 9]

    }
```

59. Few programs on Java 8 Map?

```
public class Java8MapOperations {

    public static void main(String[] args) {
        Map<Integer, String> dataMap = new HashMap<>();
        dataMap.put(101, "Alex");
        dataMap.put(103, "Bob");
        dataMap.put(104, "Dan");
        dataMap.put(102, "Carey");
        dataMap.put(106, "Carey");

        //Sort Based On Key
        Map<Integer, String> sortedBasedOnKey = dataMap.entrySet().stream().sorted(Map.Entry.comparingByKey())
                .collect(Collectors.toMap(Map.Entry::getKey, Map.Entry::getValue));
        System.out.println("sortedBasedOnKey : "+ sortedBasedOnKey); //sortedBasedOnKey : {101=Alex, 102=Carey, 103=Bob, 104=Dan}

        //Sort Based on Value
        Map<Integer, String> sortBasedOnValue = dataMap.entrySet().stream().sorted(Map.Entry.comparingByValue())
                .collect(Collectors.toMap(Map.Entry::getKey, Map.Entry::getValue));
        System.out.println("sortBasedOnValue : "+ sortBasedOnValue); //sortBasedOnValue : {101=Alex, 102=Carey, 103=Bob, 104=Dan}

        //Remove duplicate values
        Map<Integer, String> afterRemovingDuplicateValues = dataMap.entrySet().stream().filter(entry -> Collections.frequency(dataMap.values(), entry.getValue())==1)
        .collect(Collectors.toMap(Map.Entry::getKey, Map.Entry::getValue));
        System.out.println("afterRemovingDuplicateValues : "+ afterRemovingDuplicateValues); //afterRemovingDuplicateValues : {101=Alex, 103=Bob, 104=Dan}

        //Map Operations on Objects
        Map<Integer, Employee> empDataMap = new HashMap<>();
        empDataMap.put(1,new Employee(101,"Alex","Male",20000));
        empDataMap.put(3,new Employee(102,"Bob","Male",30000));
        empDataMap.put(2,new Employee(105,"Sita","FeMale",80000));
        empDataMap.put(5,new Employee(104,"Carey","Male",60000));
        empDataMap.put(4,new Employee(103,"Geeta","FeMale",10000));

        //Sort by Gender
        Map<Integer, Employee> genderMap = new LinkedHashMap<>();
        empDataMap.entrySet().stream().sorted(Map.Entry.comparingByValue(Comparator.comparing(Employee::getGender)))
        .forEach(data -> genderMap.put(data.getKey(), data.getValue()));
        System.out.println("genderMap: " +genderMap);
        //genderMap: {2=Employee [employeeId=105, name=Sita, gender=FeMale, salary=80000.0], 4=Employee [employeeId=103, name=Geeta, gender=FeMale, salary=10000.0],
        //1=Employee [employeeId=101, name=Alex, gender=Male, salary=20000.0], 3=Employee [employeeId=102, name=Bob, gender=Male, salary=30000.0],
        //5=Employee [employeeId=104, name=Carey, gender=Male, salary=60000.0]}
```

60. Find Missing number in array?

```
import java.util.stream.IntStream;

public class LogicalPrograms {

    public static void main(String[] args) {
        int[] array = {1, 2, 4, 5, 6};
        int length = array.length+1;
        System.out.println(IntStream.rangeClosed(1, length).sum());//21
         int missingNumber = IntStream.rangeClosed(1, length).sum()-IntStream.of(array).sum();
         System.out.println("missingNumber : "+ missingNumber); //missingNumber : 3
    }

}
```

61. Difference between IntStream.range and IntStream.rangeClosed?

- The range is half-open, meaning the **start** value is included, but the **end** value is excluded.
- Example:

```java
IntStream.range(1, 5).forEach(System.out::println);
```

Output:

```
1
2
3
4
```

Here, the numbers `1` through `4` are included, but `5` is excluded.

2. `IntStream.rangeClosed(startInclusive, endInclusive)`:
   - The range is fully closed, meaning both the **start** and **end** values are included.
   - Example:

```java
IntStream.rangeClosed(1, 5).forEach(System.out::println);
```

Output:

```
1
2
3
4
5
```

Here, the numbers `1` through `5` are included.